Cairo University
Faculty of Engineering
Computer Engineering
Fall 2018

Design and Analysis of Algorithms
Homework 1

DEADLINE Monday, October 1st 2018, 11:59 PM

In this assignment, it is required to implement some sorting algorithms, evaluate the performance of these algorithms and test their performance on large datasets. For this task, you will use your preferred programming language (preferably C++) to implement the sorting algorithms. After that, you will test your code using randomly generated large datasets.

Your code should be organized as follows:

1. A global function named "**Sort**" has that takes the following parameters:
   a. A list of unsorted integers. Your algorithms will be tested against lists of sizes 1000, 5000, 10000, 50000, 75000, 100000, and 500000.
   b. An identifier from 0 to 5 to refer to a certain sorting algorithm from the following list of algorithms.
      0. Selection Sort
      1. Insertion Sort
      2. Merge Sort
      3. Quick Sort
      4. Heap Sort
      5. Hybrid Sort

   This function calls the corresponding sorting function according to the identifier number. It returns the time taken by the given sorting algorithm to run.
2. A global function for each of the following sorting algorithms:
   a. selectionSort function.
   b. insertionSort function.
   c. mergeSort function.
   d. quickSort function.

e. heapSort function.
f. A hybrid algorithm of your design.

Each function should sort the passed list of unsorted integers using the corresponding sorting algorithm.

The sorting should be in an ascending order of increasing values.

The sorting is done in place, i.e. after calling each function; the passed list should be sorted.

To test the speed for the algorithm, create a timer (get the time in milliseconds) before and after the algorithm runs, and subtract the two times. Each of these functions should return the time taken in milliseconds to run the algorithm.

Try to think of a hybrid sorting algorithm that can make the sorting as efficient as possible. It's better to start by the given algorithms and compare them in order to have a clear view how to design this hybrid algorithm.

3. You will run the attached Python script to generate an unsorted list of integers with a given size. This is an automated process. You will just need to execute the following command to generate a list of 100 random integers and export them to a file named "data.txt"
```
python runscript.py 100 data.txt
```

4. In the **main** function, you will import the generated list of random integers and pass this list to the "Sort" function. The sorted data should be exported into a file named "sorted_data.txt". You should test the performance of the sorting algorithms on unsorted and sorted data with sizes 1000, 5000, 10000, 50000, 75000, 100000, and 500000. The running time of each algorithm for each list size should be exported to a file named "selection.txt", "insertion.txt", "merge.txt", "quick.txt", "heap.txt" and "hybrid.txt".
Example: The output of "selection.txt" should be as follows:
1000_unsorted: #number of milliseconds taken
1000_sorted: #number of milliseconds taken
5000_unsorted: # number of milliseconds taken

5000_sorted: # number of milliseconds taken

5. You will develop your code under Linux, so you have to run the code from terminal with the following arguments.
   a. The algorithm number.
   b. The path of the input file.
   c. The path of the output file.
   d. The path of the file to write the time to run the selected algorithm.

   The main file should be named "sort.cpp" all lowercase letters.

   Example:
   ```
   $gcc –o output sort.cpp
   --this command will build and compile sort.cpp and generate a binary
   file named output.
   $ ./output 1 data.txt sorted_data.txt running_time.txt
   --this command will run the executable generated with the given
   arguments.
   ```

<u>Deliverables:</u>

- Source code for all of your sorting algorithms
- Source code for your main program, which you used for the performance testing.
- A .pdf file containing running times for all 6 algorithms (5 standard, and 1 hybrid) for lists of 1000, 5000, 10000, 50000, 75000, 100000 and 500000. You should include sorted lists as well as random lists for each list size in these tests. Plot all results in one plot (X-axis is N, and Y-axis is time) with clear legend for the different algorithms.
- A brief (one page is enough) .pdf document on how you created your hybrid sorting algorithm, which approaches you tried, and how much of a difference these changes made to the efficiency of your algorithm.
- The code will be judged according to OO design, following a consistent coding convention, code cleanliness, and documentation.
- Remember! Plagiarism is not tolerated. Any sign of cheating or plagiarism will be graded as ZERO in this assignment and all other assignments.

Submission:

- Go to www.elearn.eng.cu.edu and login with your student account (the same you use in the student portal).
- Enroll in the course with the key sent to your class representative.
- Submit all deliverables through the website by the indicated deadline.